Training a Discriminator to Compare Generative Dialogue Models

Nicolas A.Gontier
260532513
Reasoning and Learning Lab
School of Computer Science
McGill University
Montreal, QC, Canada
nicolas.angelard-gontier@mail.mcgill.ca

Abstract

Dialogue systems have made a lot of progress recently with the use of deep learning methods. However, their evaluation remains an open problem. This project looks at one possible way to do that by investigating the notion of adversarial networks. We train a deep recurrent neural network to discriminate between true responses and generated responses. The training is done using responses from multiple generative dialogue systems with different characteristics. After analyzing our results we show that our model is able to pick on some key features that make a good dialogue response. At the same time we also conclude that such a model is not enough to judge the quality of a response as some obviously bad generation systems are often able to fool the discriminator.

1 Introduction

The goal of generative dialogue models is to provide a coherent response to some dialogue conversation history. With the rise of deep learning, recent neural architectures consisting of recurrent neural networks seems to be the state of the art in this domain (Lowe et al., 2015, 2016; Serban et al., 2015, 2016; Li et al., 2016, 2017). Overall, we can distinguish two kinds of dialogue generation models: some are retrieval based, others are purely generative based. The first will encode a context and sample an utterance from the training set that best fits the conversation according to some similarity measure (Lowe et al., 2016); the later is usually an encoder-decoder architecture where the context is encoded and the decoder outputs the next tokens based on the likelihood of the training data (Serban et al., 2016). Both of these methods have some advantages and weak points as we will see in our analysis.

Unlike in goal-oriented conversations, open domain dialogues don't have a clear definition of "success" and their evaluation remains an open question. The solution to this problem forces us to think about what makes a *good* conversation and how to automatically measure it. The automation here is very important as human measurement is expensive and does not scale to big domains. Liu et al. (2016) showed that N-gram metrics such as *ROUGE* and *BLEU* should be avoided for the dialogue task as they correlate very poorly with human judgment. Recently, Li et al. (2016) manually defined three properties that measure the quality of an utterance such as ease of answering, informativeness and coherence. However, these metrics remain heuristics and cannot cover the wide spectrum of all conversational aspects. Alternatively, Lowe et al. (2017) build a scoring machine on human labeled data and showed that their correlation was better than *BLEU* and *ROUGE* scores. Here again, the authors argue that this metric is only a first step towards dialogue evaluation and that there is still a lot do be done.

This work uses a discriminator network for the dialogue evaluation task. The goal of such a model is to learn how to discriminate between "true" responses and generated responses for some conversation history. This project compares the accuracy of our discriminator against different dialogue models. Following the Turing test reasoning, we believe that a good generated dialogue response should look similar to a human one, and thus should fool the discriminator network. In fact, we show that this idea is not always true and that fooling the adversarial network cannot be enough to conclude on the performance of a generative dialogue system.

2 Models Description

Recurrent Neural Networks (*RNNs*) are a variant of neural networks that is specially well suited for time-series data like text. At each time steps t one token x_t is fed into the network and the hidden state h_t is updated:

$$h_t = f(W_h h_{t-1} + W_x x_t) \tag{1}$$

where h_{t-1} is the previous hidden state, W_h and W_x are weight matrices applied to the the previous hidden state and the input respectively. One common issue with *RNNs* is the problem of exploding or vanishing gradients. This happens when the error gradient signal increases or decreases exponentially over time. In order to avoid this we use long short term memory (*LSTM*) units (Hochreiter and Schmidhuber, 1997). These type of network units have been shown to better model long time-dependent series than standard *RNNs*(Chung et al., 2014). To further address the issue of exploding gradients, we also utilize gradient clipping.

2.1 Discriminator Network

Our discriminator network is a dual encoder architecture where we use two *RNNs* with tied weights to encode a context–response pair. Given a *(context, response, flag)* triple where *flag* is a binary token indicating if the response is true to its context or not, we compute the context and response encodings by feeding them into their respective network. At each time step, we pass the current word embedding to the *LSTM* unit and update the current hidden state. Word embeddings are initialized randomly and trained by the network. The last hidden state of each *LSTM* network is a vector representation of its input and can be though of a summary of the tokens we have seen. Once we obtained our context and response encodings $c \in \mathbb{R}^n$ and $r \in \mathbb{R}^n$ respectively, we compute the probability that this is a valid pair:

$$P(flag = 1|c, r, M) = \sigma(c^T M r) \tag{2}$$

where the weight matrix $M \in \mathbb{R}^{n \times n}$ is learned by the network and can be seen as a similarity measure between the context and the response. Eventually, the sigmoid function converts the result to a probability. The predicted label $\hat{y}$ is 1 if the probability is greater than 0.5, and 0 otherwise. An example of our architecture can be seen in Figure 1.

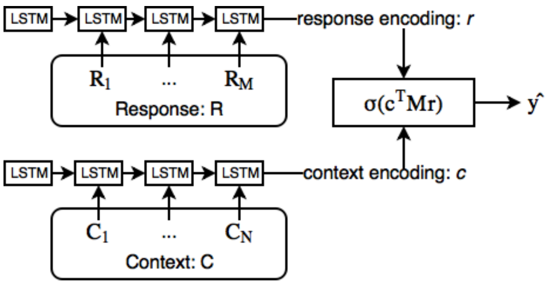

Figure 1: Discriminator network

We train the network by minimizing the binary cross entropy loss between our prediction $\hat{y}$ and the true *flag* (denoted y) for each context–response pair within a training batch:

$$J(\hat{y}, y) = -y * \log P(\hat{y}) - (1-y) * \log(1 - P(\hat{y})) \tag{3}$$

To train the network we take the gradient of this objective function with respect to the network parameters and perform stochastic gradient descent. We detail our implementation method in section 3.2

2.2 Generative Dialogue Models

In order to train our network, we sampled from the following generative models and flagged their response as ‘false’ examples for the discriminator, so setting $y = 0$ for each sampled response.

Random: This is a retrieval based system which assigns a random response to each context. Since responses are sampled we know that their syntax is identical to the true human style, however we expect their meaning to be wrong for the given context. A discriminator network that understand the meaning of a conversation should be able to flag these response as false.

TF-IDF: We consider a slightly less naive retrieval based model that relies on term frequency-inverse document frequency (Lowe et al., 2015). This method samples the response that is most similar to a given context based on the number of time its tokens appear in the context. Being a retrieval model we have the same assumptions as for the *random* model, but with some improvement in terms of coherence accuracy.

HRED: We now consider generative based models with a hierarchical recurrent encoder decoder (Serban et al., 2015) that outputs new responses for each given context. The hierarchical encoder gives a vector representation of the context, and the decoder produces a probability distribution over the set of tokens to output. Within this model we consider two different ways of sampling from the decoder distribution: one is stochastic, the other uses beam search with a beam of size 5. The first sampling method simply picks a token according to the decoder distribution at each time step. The later essentially restrains its sampling horizon by picking only very probable tokens at each time step. While the beam sampling is expected to return "higher quality" responses, the stochastic sampling will cover a wider range of behavior and might be less obvious to classify for our discriminator.

VHRED: Eventually, we also consider a hierarchical latent variable recurrent encoder decoder (Serban et al., 2016) as another generative system. This model has the same hierarchical encoder as the *HRED* model, but differs in its decoder. Here the decoder is conditioned on the encoded context as well as on hidden Gaussian latent variables. The network first samples the latent variable based on the context encoding and then starts generating output distributions conditioned on the latent variable. This has been shown to produce better responses than the *HRED* model so we expect our discriminator network to be weaker on *VHRED* responses. Here again we consider two types of sampling: stochastic and beam search of size 5.

3 Methodology

In this section we present the data we used and provide hyper-parameter settings.

3.1 Data description

For this project we used the Twitter corpus collected following the procedure of Ritter et al. (2010), where the dialogues are usually "chit-chat" between two or three users with no particular conversation goal. The original size of the data vocabulary was around $20,000$ words, so for efficiency reasons we applied *Byte Pair Encoding* (*BPE*) to reduce the vocabulary size to $5,000$ tokens. *BPE* was applied to natural language as a way to generate sub-word level tokens (Sennrich et al., 2015).

The data set is composed of $749,060$ context – response pairs. We flagged the original responses as 'true' examples ($y = 1$). For each contexts we also sampled responses from the different pre-trained dialogue models described in section 2.2 and flagged their responses as 'false' ($y = 0$). Note that each of these generative models were previously trained on the same twitter data set, thus making them robust on this domain. It is also important to note that we do not alter these models in any way once they are pre-trained. We simply use them as data generators for training our discriminator. Sampling responses from each of those resulted in $6 \times 749,060 = 4,494,360$ 'false' responses ($y = 0$) for our discriminator. Some dialogue examples can be found in Appendix A.

After dividing the original $749,060$ contexts into 80% train, 10% validation, and 10% test set, we added the true responses and the 6 types of false responses (*random*, *tf-idf*, *HRED-stochastic*, *HRED-beam5*, *VHRED-stochastic*, *VHRED-beam5*) for each context. Eventually we oversampled true responses in the training set to have the same amount of positive ($y = 1$) and negative ($y = 0$) training examples in order to avoid the class-imbalance problem. The final amount of data was $7,190,976$ training, $524,343$ validation and $524,343$ test examples.

3.2 Implementation details[1]

This work was coded in *Python* using the *Theano* library (Theano Development Team, 2016) along with the *Lasagne* package (Dieleman et al., 2015). We used cross-validation on the validation set to pick the best set of parameters for this task. In order to optimize our objective function described in Equation 3 we exploit the *ADAM* optimizer (Kingma and Ba, 2014) as it yield better accuracies on the validation set. The learning rate was set to 0.001 with an exponential decay rate for the first moment estimate of 0.9 and of 0.999 for the second moment estimate. The dimension of our vector representation of the hidden state (ie: the number of hidden nodes) was set to 100. Note that we used tied weights between the context recurrent network and the response recurrent network. This was done to optimize the training time of our algorithm as it requires twice as less parameters. We clipped our gradients to a maximal value of 10 and our probabilities to be in the following range: $[10^{-7}, 1 - 10^{-7}]$. We initialized our word embeddings to random float vectors of size 300 with values between 0 and 1. The matrix M from Equation 2 was initialized to the identity. Eventually, sequences longer than 160 tokens were truncated, and we trained our model up to a patience of 5: if the average validation score over all models was

[1] The code is available at
`https://github.com/NicolasAG/Discriminator`

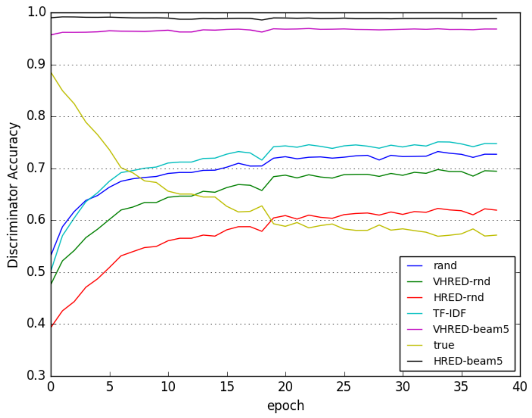

Figure 2: Discriminator accuracies on validation set

Model	True negative	False positive
Random	73%	27%
TF-IDF	75%	25%
HRED (random)	63%	37%
HRED (beam-5)	99%	1%
VHRED (random)	70%	30%
VHRED (beam-5)	97%	3%

True positive	False negative
57%	43%

Table 1: Discriminator test set accuracy

not improved within 5 epochs we stopped training and assumed our model has converged.

4 Discussion

After training our algorithm on this large data set and with the implementation described above we report the validation learning curve of our discriminative model on each generative model in Figure 2. It is important to understand that lower curves represent models that can more easily fool the discriminator.

To our surprise, the discriminator network can very easily identify responses from generative state of the art models like *HRED*, and *VHRED* when sampled with beam search (black and purple curves respectively). In addition, we can see that it can do so with very minimal learning. This suggests that our network is able to pick on some human dialogue characteristics that are clearly missing from these models. Note that this is not the case in the random sampling case (red and green curves). We should first mention that the big difference between random sampling and beam sampling is probably due to the fact that having a beam size of 5 greatly reduces the space of actions that our generative model can use to fool the discriminator. On the other hand, having a stochastic sampling makes the generator less predictable and thus harder to discriminate.

Moreover, the fact that using naive retrieval based methods like the *random* model (blue curve) or the *tf-idf* model (cyan curve) is able to fool the trained discriminator roughly 30% of the time is a bad result for our adversarial model. As we can see in Appendix A, humans can easily find problems with these models, yet the discriminator can't. This suggest that our network is looking very much at superficial features of the responses such as the length and overall syntactic structure of the dialogue.

In addition, the fact that the dialogue models that best fool the discriminator are *HRED* and *VHRED* with random stochastic sampling (red and green curves respectively) suggests that the adversarial network is slightly looking at the dialogue coherence. Indeed, even though these models are not the best from our human judgment, they are still better (and harder to discriminate) than a *random* or a *tf-idf* sampler.

The other surprising result that we can see from Figure 2 is that the ability to identify actual true responses decrease over time (yellow curve). Since we made sure to have the same amount of true ($y = 1$) and false ($y = 0$) examples in our training set this is not a class-imbalance problem. Thus this result should be taken as a warning when using adversarial networks in the dialogue domain.

Eventually, we also report the accuracy scores of our network on the hold out test set in Table 1. true negatives represent the percentage of time the discriminator correctly flagged the response as false, while false positive represent the percentage of time the discriminator was fooled by the dialogue model and though it was a true response. We also show the discriminator's final accuracy at labeling true responses. As discussed previously, the true positive percentage was higher at the beginning of training.

5 Conclusion

In this project we presented an automated way of evaluating dialogue responses by using an adversarial network. Our results let us conclude that even though an adversarial method may give some feedback, it is definitely not as “smart” as a human evaluation. First of all, we showed that the event of fooling an adversarial network does not allow to conclude that a dialogue model is ‘good’. In addition we also showed that a discriminator with high true negatives may be at the expense of low true positives. We believe that the dialogue evaluation task requires more than one model, and should really be a combination of multiple ones. As future work, our adversarial network could be improved by using pre-trained word embeddings like word-2-vec, but also pre-trained utterance embedding after training a generative encoder-decoder network on likelihood. Moreover, we also plan to investigate the idea of training multiple adversarial networks to each discriminate on one different model. We think that the use of transfer learning between all of these will give us better results on new context–response pair.

Acknowledgments

The authors would like to thank Iulian V. Serban for providing the model responses, as well as Michael Noseworthy and Ryan Lowe for useful discussions and insights. We gratefully acknowledge financial support for this work by the Samsung Advanced Institute of Technology (SAIT).

References

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. https://arxiv.org/abs/1412.3555.

Sander Dieleman, Jan Schlter, Colin Raffel, Eben Olson, Sren Kaae Snderby, Daniel Nouri, et al. 2015. Lasagne: First release. https://doi.org/10.5281/zenodo.27878.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. https://arxiv.org/abs/1412.6980.

Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. 2016. Deep reinforcement learning for dialogue generation. https://arxiv.org/abs/1606.01541.

Jiwei Li, Will Monroe, Tianlin Shi, Sbastien Jean, Alan Ritter, and Dan Jurafsky. 2017. Adversarial learning for neural dialogue generation. https://arxiv.org/abs/1701.06547.

Chia-Wei Liu, Ryan Lowe, Iulian V. Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. https://arxiv.org/abs/1603.08023.

Ryan Lowe, Michael Noseworthy, Iulian V. Serban, Nicolas A.Gontier, Yoshua Bengio, and Joelle Pineau. 2017. Towards an automatic turing test: Learning to evaluate dialogue responses.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. https://arxiv.org/abs/1506.08909.

Ryan Lowe, Iulian V. Serban, Mike Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. On the evaluation of dialogue systems with next utterance classification. https://arxiv.org/abs/1605.05414.

Alan Ritter, Colin Cherry, and Bill Dolan. 2010. Unsupervised modeling of twitter conversations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, HLT ’10, pages 172–180. http://dl.acm.org/citation.cfm?id=1857999.1858019.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. https://arxiv.org/pdf/1508.07909.pdf.

Iulian V. Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2015. Building end-to-end dialogue systems using generative hierarchical neural network models. https://arxiv.org/abs/1507.04808.

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2016. A hierarchical latent variable encoder-decoder model for generating dialogues. https://arxiv.org/abs/1605.06069.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints* abs/1605.02688. http://arxiv.org/abs/1605.02688.

A Dialogue Examples

Context: A: wtf. B: what's wrong? are you okay?
True: A: i'm okay just kinda frustrated. will explain later!
Random: A: looking foward to it homie u need to come hard!
TF-IDF: B: it probably not you hun
HRED-rnd: A: yes. sadly not. but it's ok i'm really not gonna ruin it for myself. but there's no chance it felt weird.
HRED-beam5: A: yeah, i'm fine.
VHRED-rnd: A: i'm good.
VHRED-beam5: A: i'm fine. just tired.

Table 2: Example 1 of multiple responses to a specific context

Context: A: i'm watching make-up tutorials on youtube. i reallyyyy wanna learn how to wear make-up!!! B: o wow i knew u would come around one day
True: A: lmfao yeah i know. but i want a natural look doe!!! nothing to much!
Random: B: i'm barely going we went to pick up evelyn.
TF-IDF: A: aha xd sure! next thursday we'll see what a manicure guru you are
HRED-rnd: C: ws da power beat
HRED-beam5: A: i know!!!!!!!
VHRED-rnd: B: read it
VHRED-beam5: A: i'm not going to school tomorrow. i'm not going to school tomorrow.

Table 3: Example 2 of multiple responses to a specific context